LISTING OF THE CLAIMS

Claims 1-16 are pending. Claims 1, 7 and 12 are amended. Claims 2, 8 and 13 are canceled. The remaining claims are unchanged.

1. (Currently Amended) A computer system comprising:

a preloader arranged to,

determine whether a bytecode makes an active reference to a class which requires an execution of a static initializer,

determine if the class has a superclass which requires the execution of the static initializer, wherein the preloader produces a source file[[;]],

rewrite the bytecode to a new bytecode which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer;

a compiler coupled to the preloader arranged to accept the source file as input and produce an object file; and

a virtual machine coupled to the compiler arranged to execute the object file.

2. (Canceled).

3. (Original) A computer system according to claim 1 wherein the preloader is further arranged to:

rewrite the bytecode to a new bytecode which explicitly indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the

bytecode makes the active reference to the class which requires the execution of the static initializer.

4. (Original) A computer system according to claim 1 wherein the preloader is further arranged to:

rewrite the bytecode to a new bytecode which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which has the superclass which requires the execution of the static initializer.

5. (Original) A computer system according to claim 1 wherein the preloader is further arranged to:

rewrite the bytecode to a new bytecode which explicitly indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which has the superclass which requires the execution of the static initializer.

6. (Original) A computer system comprising:

a bytecode rewritter arranged to,

determine whether a bytecode is associated with a scalar field or an object reference field,

rewrite the bytecode to identify the bytecode as being associated with the scalar field when the bytecode is associated with the scalar field,

rewrite the bytecode to identify the bytecode as being associated with the object reference field when the bytecode is associated with the object reference field, wherein the bytecode rewriter is associated with producing a source file;

a compiler arranged to accept the source file as input and produce an object file; and

a virtual machine arranged to execute the object file.

7. (Currently Amended) In a computer system having a preloader coupled to a compiler and a virtual machine, a method for rewriting bytecodes to minimize runtime checks, comprising:

by the preloader,

determining whether a bytecode makes an active reference to a class which requires an execution of a static initializer;

determining if the class has a superclass which requires the execution of the static initializer, wherein the preloader produces a source file**[[;]]**;

rewriting the bytecode to a new bytecode, by the preloader, which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer;

accepting the source file as input and produce an object file by the compiler; and

executing the object file by the virtual machine.

8. (Canceled).

9. (Original) A method according to claim 7, further comprising:

rewriting the bytecode to a new bytecode, by the preloader, which explicitly indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer.

10. (Original) A method according to claim 7 further comprising:

rewriting the bytecode to a new bytecode, by the preloader, which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which has the superclass which requires the execution of the static initializer.

11. (Original) A method according to claim 7, further comprising:

rewriting the bytecode to a new bytecode, by the preloader, which explicitly indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which has the superclass which requires the execution of the static initializer.

12. (Currently Amended) A computer program product for rewriting bytecodes to minimize runtime checks in a computer system having a preloader coupled to a compiler and a virtual machine, comprising:

computer code for determining whether a bytecode makes an active reference to a class which requires an execution of a static initializer;

computer code for determining if the class has a superclass which requires the execution of the static initializer, wherein the preloader produces a source file;

computer code for rewriting the bytecode to a new bytecode, by the preloader, which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer;

computer code for accepting the source file as input and produce an object file by the compiler;

computer code for executing the object file by the virtual machine; and

a computer readable medium for storing the computer program product.

13. (Canceled).

14. (Original) A computer program product according to claim 12, further comprising:

computer code for rewriting the bytecode to a new bytecode, by the preloader, which explicitly indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer.

15. (Original) A computer program product according to claim 12 further comprising:

computer code for rewriting the bytecode to a new bytecode, by the preloader, which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which has the superclass which requires the execution of the static initializer.

16. (Original) A computer program product according to claim 12, further comprising:

computer code for rewriting the bytecode to a new bytecode, by the preloader, which explicitly indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which has the superclass which requires the execution of the static initializer.